[WR 2001.015; DDK 218.1023]

TEXT STREAM FILTER

INVENTOR:
Christian Linhart

PREPARED BY:

Davidson, Davidson & Kappel, LLC
485 Seventh Avenue
New York, N.Y. 10018
212-736-1940

TEXT STREAM FILTER

BACKGROUND INFORMATION

[0001] In the computer science field, text stream filters find application in a variety of applications, such as parser filters, pattern matchers, etc. Such filters may be used to modify a text stream being inputted to a device so as to achieve desired properties of the text stream. For example, a text stream filter may be used to filter input to a parser, thus serving as a parser filter.

[0002] A parser is a program for extracting syntactic, symbolic and/or semantic information from source code. A typical parser includes a front end, usually a scanner, for tokenizing an input source code stream. The scanner compares the input source code stream to a set of predefined patterns. When a pattern is matched, an action which has been defined for the pattern is executed. For example, the matched source code may be sent as a token to a syntax analysis subsystem. Scanners may be stand-alone programs or included in other applications, such as compiler front ends, text editors for syntax highlighting, and text filters.

[0003] A parser filter may be used upstream of a parser to modify the input to the parser so that the parser can still process source code containing nonstandard code. Such modification may be useful, for example, for parsing a dialect of a language. For example, a parser filter might implement "ignore" rules to replace with blanks certain nonstandard text strings which might not be understandable by the parser.

[0004] Prior text filters have a number of limitations. First, they are capable of replacing input text only by blanks, or whitespace, ("ignoring") but not by other text ("replacing"). Prior text filters do not provide positioning information indicating, for example, the length of an original text block before a replacing action. As a consequence, ignore rules must often be defined to ignore certain source code constructs completely. Significant amounts of source code may not be read by the

parser, leading to missing symbols. Without positioning information, input text may only be replaced with replacement strings having the same length as the text to be replaced. When only ignore rules are used, matched text may be replaced with an equal number of whitespace characters. To perform replacements of arbitrary length, positioning information is necessary.

[0005] Secondly, prior text filters do not support context-dependent patterns, i.e., they provide no mechanism for controlling in which situations an ignore or replace rule is active. Additionally, prior text filters do not support full regular expressions as patterns. Furthermore, prior text filters used as parser filters are interwoven tightly in a specific parser's code, so for every parser a tailored parser filter is required.

[0006] Thus, prior text filters are relatively inflexible.

SUMMARY

[0007] In accordance with a first embodiment of the present invention, a method for filtering an input to a parser is provided. The method includes receiving a definition of a filter configuration and modifying the input text stream according to the filter configuration so as to generate a filtered text stream including positioning information for the input text stream.

[0008] In accordance with a second embodiment of the present invention, an apparatus for filtering a text stream is provided. The apparatus includes a filter configuration data structure and a scanner unit configured to modify the input text stream using the filter configuration data structure so as to generate a filtered text stream, the filtered text stream including positioning information for the input text stream.

[0009] In accordance with a third embodiment of the present invention, a computer readable medium having stored thereon computer executable process steps operative to perform a method for filtering an input text stream is provided. The method includes:

receiving a definition of a filter configuration; and modifying the input text stream according to the filter configuration so as to generate a filtered text stream, the filtered text stream including positioning information for the input text stream.

[0010] In accordance with a fourth embodiment of the present invention, a parsing device is provided. The parsing device includes: a filter configuration data structure; a first scanner unit configured to modify the input text stream using the filter configuration data structure so as to generate a filtered text stream, the filtered text stream including positioning information for the input text stream; and a second scanning unit configured to process the filtered text stream so as to apply the positioning information.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] Fig. 1 shows a flow chart of a method for filtering an input text stream.

[0012] Fig. 2 shows a schematic diagram demonstrating filtering of a text stream.

[0013] Fig. 3 shows a schematic block diagram for filtering an input text stream to a parser.

[0014] Fig. 4 shows a flow chart of a method for generating a scanner .

[0015] Fig. 5 shows a flow chart of a method for generating a scanner using start states associated with patterns to be scanned for.

DETAILED DESCRIPTION

[0016] As described above, the present invention provides a method for filtering an input text stream. The filtered text stream produced includes positioning information for the input text stream. This enables text replacements, in addition to text ignores, to be performed by the filter. The method in accordance with certain embodiments of the

present invention includes receiving a definition of a filter configuration, and modifying the input text stream according to the filter configuration so as to generate a filtered text stream including positioning information for the input text stream.

[0017] Receiving a definition of the filter configuration may include receiving a definition of a plurality of patterns and receiving a definition of a respective association between each of the plurality of patterns and a respective executable action. Moreover, receiving a definition of the filter configuration may include processing each of the plurality of patterns and the respective association so as to form a scanner data structure capable of comparing the input text stream to at least one of the plurality of patterns, and causing execution of the associated executable action upon a match of the input text stream with the respective one of the plurality of patterns. The processing and the comparing may be performed in a same active process.

[0018] Receiving a definition of the filter configuration may further include associating at least one respective start state of a plurality of start states with each of the plurality of patterns, and setting a current start state, the current start state being one of the plurality of start states. The processing may include processing the at least one respective start state along with each associated pattern so as to form a part of the scanner data structure. The comparing may be performed so as to compare only the patterns of the plurality of patterns associated with a respective start state equal to the current start state. Moreover, each respective start state is included in a context.

[0019] The definition of the filter configuration may be received from a data structure.

[0020] At least one of the respective executable actions m ay include replacing a matched text in the input text stream with a respective replacement text. Moreover, modifying the input text stream may include replacing at least one character in the input text stream with a replacement text. The replacement text may be determined dynamically, may be computed by the respective executable action, and/or determined

using at least a portion of the matched text.

[0021] The method according to the present invention may further include comparing the input text stream to a plurality of patterns and causing replacing of matched text in the input text stream with a respective replacement text upon a match of the input text stream with the respective one of the plurality of patterns, the replacing being included in at least one of the respective executable actions.

[0022] Modifying the input text stream may include determining a difference in a number of characters between the at least one character and the replacement text. The positioning information may include the difference.

[0023] The positioning information may refer to positions of characters in the input text stream. Moreover, the positioning information may include directives. The directives may include executable actions, the executable actions being executable by a scanner receiving the filtered text stream.

[0024] Certain embodiments of the method according to the present invention may further include providing a modified scanner generator skeleton configured to receive the filtered text stream, the modified scanner generator skeleton being useable to generate a scanner capable of applying the positioning information. Moreover, the method according to the invention may include generating the scanner using the modified scanner generator skeleton. The scanner may form part of a parser. The part may include a front end of the parser.

[0025] Receiving a definition of the filter configuration may further include associating at least one respective start state of a plurality of start states with each of the plurality of patterns, and setting a current start state, the current start state being one of the plurality of start states. The processing may include processing the at least one respective start state along with each associated pattern so as to form a part of the

scanner data structure. The comparing may be performed so as to compare only the patterns of the plurality of patterns associated with a respective start state equal to the current start state. Moreover, each respective start state is included in a context.

[0026] The present invention also provides an apparatus for filtering a text stream. The apparatus includes a filter configuration data structure and a scanner unit configured to modify the input text stream using to the filter configuration data structure so as to generate a filtered text stream. The filtered text stream includes positioning information for the input text stream.

[0027] The present invention also provides a computer readable medium having stored thereon computer executable process steps operative to perform a method for filtering an input text stream. The method includes receiving a definition of a filter configuration, and modifying the input text stream according to the filter configuration so as to generate a filtered text stream. The filtered text stream includes positioning information for the input text stream.

[0028] The present invention also provides a parsing device including a filter configuration data structure and a first scanner unit configured to modify the input text stream using the filter configuration data structure so as to generate a filtered text stream. The filtered text stream includes positioning information for the input text stream. The parsing device also includes a second scanning unit configured to process the filtered text stream so as to apply the positioning information. In certain embodiments, the second scanning unit may include a modified scanner generator skeleton configured to receive the filtered text stream. The modified scanner generator skeleton may be useable to generate a scanner capable of applying the positioning information.

[0029] Fig. 1 shows a flow chart of a method according to the present invention for filtering an input text stream. First, a filter configuration is defined (Step 102). Then,

the input text stream is modified according to the filter configuration so as to generate a filtered text stream, the filtered text stream including positioning information for the input text stream (Step 104).

[0030] Fig. 2 shows a schematic diagram demonstrating filtering of a text stream according to the present invention. Input text stream 202 includes a serial stream of input text strings A, B, C. Only three text strings are depicted, though more or fewer text strings could be present in text stream 202. Input text strings A, B, C may each include one or more text characters. Input text stream 202 is input into text stream filter 204. Text stream filter 204 processes input text stream 202 based on filter configuration 205, and outputs filtered text stream 206. The break in arrow 203 between filter configuration 205 and text stream filter 204 in Fig. 2 indicates that additional processing may be performed on filter configuration 205 before it is inputted into text stream filter 204. Text stream filter 204 searches for predefined text patterns in input text stream 202 and, upon a match, performs a respective predefined action on the matched text. Text stream filter 204 also produces position information for text strings A, B, C and interleaves the position information into filtered text stream 206.

[0031] Filter configuration 205 includes one or more rules for filtering input text stream 202. The rules may include text patterns, i.e., regular expressions, text, or other arrangements of characters, symbols, etc., to be searched for in input text stream 202, and start states used for controlling when a given pattern is active, i.e., when input text stream 202 is to be searched for the given pattern--and when not. Alternatively, "contexts" may be used to control when a given pattern is active. Contexts are described in the patent application entitled "Dynamic Scanner," applicant docket number 218.1022, assigned to applicant and filed on even date herewith, and which is herewith incorporated by reference herein. Additionally, the rules for filtering input text stream 202 may include actions to be performed upon a match with one of the patterns. The actions may include replacing the matched text in the input stream with another text, replacing the matched text with blank characters, or simply copying the

matched text to filtered text stream 206.

[0032] Filtered text stream 206 includes a serial stream of text-directive pairs 208. Only three text-directive pairs are depicted, though more or fewer number pairs--corresponding to the number of input text strings A, B, C in input text stream 202--could be present in text stream 202. Text-directive pairs 208 each include a respective output text string A', B', C' and a respective associated directive P_A, P_B, P_C.

[0033] Each output text string A', B', C' may include either: a copy of the corresponding input text string A, B, C; text inserted by text stream filter 204 as a replacement for the corresponding text string A, B, C; an empty string inserted by text stream filter 204 as a replacement for the corresponding input text string A, B, C (pursuant to an ignore rule).

[0034] Directive P_A, P_B, P_C provide positioning information. Each directive P_A, P_B, P_C includes a number representing the difference between the number of characters in each output text string A', B', C' and in the respective input text string A, B, C. In other embodiments of the present invention, other types of positioning information may be used.

[0035] Text stream filter 204 may include a scanner generated using a scanner generator, such as Flex, for example. Flex is a scanner generator developed as open-source software by the University of California at Berkeley. Alternatively, a scanner generated by any suitable scanner generator may be used. The scanner may perform the pattern-matching and associated replacement, etc., actions on input text stream 202. The scanner may be a scanner generated by inputting patterns, start states and actions into an appropriate scanner generator as a scanner definition file, i.e., a filter configuration file. The scanner generator is run, and a scanner in the form of source code including serialized data structures the actions from the scanner definition file verbatim. After compiling, the scanner in the form of a scanner data structure in

machine language is ready to be run for text filtering operations as a stand-alone scanner or linked to object code of another program. Such a scanner may be referred to as a static scanner.

[0036] In other embodiments of the present invention, text stream filter 204 may include a dynamic scanner as described in the "Dynamic Scanner" application of applicant, referenced above. Using the dynamic scanner for text stream filter 204 enables the text stream filter itself to be generated without intermediate compiling being performed between forming the filter configuration file and outputting of the finished text stream filter. The dynamic scanner may include a data structure which includes, among other things, pointers to action objects for performing the replacement, etc., actions on input text stream 202.

[0037] Filtered text stream 206 may be fed into a suitable client or other device (not shown) which is capable of interpreting text-directive pairs 208. Such a client may include a parser front end, as described with reference to Fig. 3.

[0038] Fig. 3 shows a schematic block diagram according to an embodiment of the present invention for filtering input text stream to a parser. Parser filter 312 filters input text stream 314 and outputs filtered text stream 316. Input text stream 314 may be, for example, a stream of source code to be parsed. Filtered text stream 316 includes positioning information 317 for input text stream 314. Preferably, filtered text stream 316 includes text-directive pairs, each text-directive pair including a text portion and a number representing positioning information for input text stream 314, as described above with reference to input text stream 202 in Fig. 2. Filtered text stream 316 is fed into parser front end 318.

[0039] To form parser filter 312, parser filter configuration file 302 is formed. Parser filter configuration file 302 includes patterns to be searched for input text stream 314, one or more start states associated with each pattern for controlling when the pattern is

active (to be searched), and executable actions to be performed upon a matching of a pattern in input text stream 314. The actions may include replacement of at least one character of replacement text in input text stream 314 with other text, replacement of text in the input text stream 314 with an empty string, i.e., an "ignore action," or simply copying the matched text to filtered text stream 316.

[0040] The replacement text may be determined dynamically, for example, using the matched text in input text stream 314. Examples of determining the replacement text using the matched text include: the replacement text is matched text converted to upper-case characters; the replacement text is matched text converted to lower-case characters; the replacement text includes parts of the matched text which are matched by sub-expressions of a regular expression used to match the matched text; the replacement text is an altered sequence of the characters and/or sub-strings of the matched text; and combinations of the above.

[0041] Parser filter configuration file 302 is input to parser configuration interpreter 304, which converts the file to parser filter configuration data structure 306. Parser configuration interpreter 304 is a parser which parses the parser configuration file and generates a parser filter configuration data structure. Parser configuration interpreter 304 may include a dynamic scanner according to an embodiment of the invention described in the application of applicant entitled "Dynamic Scanner," referenced above. Moreover, parser configuration interpreter 304 may include a syntax analysis device generated by a LALR(1) parser generator, such as YACC, BISON, LARK, etc. Parser filter configuration data structure 306 forms an input to parser filter generator 308. Parser filter configuration data structure 306 preferably has a format appropriate for being processed by parser filter generator 308.

[0042] Parser filter generator 308 may be a dynamic scanner generator as described in the "Dynamic Scanner" application of applicant, referenced above. Alternatively, parser filter generator 308 may be a scanner generator, such as Flex, for example.

Parser filter generator 308 outputs processed parser filter generator definition 310, which includes scanning device 311. Scanning device 311 may be a dynamic scanner data structure as described in the "Dynamic Scanner" application of applicant, referenced above. Such a dynamic scanner data structure may include pointers to action objects for performing the actions defined in parser filter configuration file 302. Alternatively, scanning device 311 may be a static scanner in the form of source code, produced by Flex, for example. Such a static scanner would require compiling before use in parser filter 312.

[0043] Parser filter 312 processes input text stream 314 using processed parser filter definition 310. Searching for text matching the patterns defined in parser filter configuration file 302 and performing the corresponding actions when a match is found is performed by scanning device 311. Positioning information is determined by the executable actions which perform the text replacements upon a matching of a pattern in input text stream 314. An executable action may generate one or more directives. A directive may include the difference in length between the matched text and the replacement text, as described above. An executable action has access to both the matched text and the replacement text for determining this difference in length.

[0044] Cascading of multiple parser filters may be achieved by using an embodiment of the dynamic scanner described in the "Dynamic Scanner" application of applicant, referenced above. Scanning device 311 of processed parser filter definition 310 may be a dynamic scanner. The dynamic scanner skeleton may be modified so that the dynamic scanner can read a text stream with positioning information. The input and output of parser filter 312 then have the same format, enabling the parser filter to be cascaded with other similar parser filters.

[0045] Filtered text stream 316 is fed into parser front end 318. Parser front end 318 processes filtered text stream 316 so as to extract and interpret the positioning information included therein. Parser front end 318 may include scanner 320 for

extracting and interpreting the positioning information. Scanner 320 may include scanner skeleton 322, which may be a modified scanner skeleton of a scanner generator such as Flex or Rex, for example. The scanner skeleton is here intended to include source code which is nearly invariant over all scanners generated by the respective scanner generator, i.e., the source code which is independent of the scanner configuration--patterns, start states and actions. In addition, the scanner skeleton includes indicators on where to insert source code which depends on the scanner definition. The scanner skeleton may also include source code present in the generated scanner when certain types of patterns or actions are used.

[0046] Parser front end 318 may be implemented on top of the Parser Framework of the SNiFF+™ integrated development environment of applicant Wind River Systems, Inc. Scanner skeleton 322 is then present as a part of the Parser Framework.

[0047] Scanner 320 may be generated using scanner skeleton 322. Scanner 320 may be a static scanner or it may be an implementation of the dynamic scanner as described in the "Dynamic Scanner" application of applicant, referenced above. Employing the dynamic scanner in parser front end 318 permits positioning information 317 to be applied automatically in parser front end 318 without modification of the parser front end by an author of rules for parser front end 318. Using the dynamic scanner enables a run-time configurable parser, useful, for example, for parsing of multiple, diverse dialects.

[0048] Positioning information 317 may include directives. The directives may be implemented as executable actions, or action-objects, which are executable by scanner 320 to modify parser front end 318. The action objects may include a set of action-functions tailored to the particular type of scanner 320. Scanner 320 may include variables which indicate the position of the currently scanned part of input text stream 314. The directives change the values of these variables such that the variables indicate the corresponding position in the original input text stream.

[0049] Reference may now be had additionally to Fig. 4. Fig. 4 shows a flow chart of a method for generating a scanner in accordance with an embodiment of the invention described in the application of applicant entitled "Dynamic Scanner,"referenced above. The scanner generated may, for example, be used in parser filter 312 for filtering input text stream 314, or may serve as scanner 320 in parser front end 318. First a plurality of patterns is defined (Step 402). The patterns may include regular expressions, text patterns, or other arrangements of characters, symbols, etc., for which it is desired to search input data, such input text stream 314. The patterns may be defined by entering each pattern into a file, table or array in a memory structure, for example, such as parser configuration file 302. The input data may be a data stream, data file, or any suitable data which can be read in.

[0050] Next, a respective association between each of the plurality of patterns and a respective executable action is defined (Step 404). Each association may take the form of a pointer, an action-pointer, an index, or any suitable means of associating a pattern with a respective executable action. The receiving a definition of the associations may be performed in the same active process as the processing of the patterns and associations, as described below. Each executable action may include a respective action object, program code and/or data. In some embodiments of the present invention, an action object may be generated in the same active process as the active process in which the patterns and associations are processed, as described below.

[0051] Each of the plurality of patterns and the respective association are then processed so as to form a scanner data structure capable of comparing input data to each of the plurality of patterns and causing execution of the associated executable action upon a match of the input data with the respective one of the plurality of patterns, with the processing and the comparing being performed in the same active process (Step 406). As noted above, the receiving a definition of and processing of the patterns and associated executable actions may occur in the same active process. Additionally,

when the executable action includes an action object, the action object may be generated in the same active process. The same "active process" is here understood to mean a same running process with no intervening compiling, exiting to other programs, etc.

[0052] The patterns and respective associations with executable actions are preferably arranged, or loaded, into a scanner definition data structure. The scanner definition data structure may contain information analogous to the information contained in the scanner definition file for a prior art scanner generator, i.e., start states, patterns with respective associated actions and respective sets of the start states. Preferably, however, according to the present invention the actions are represented in the scanner definition data structure by a pointer to an action-object residing elsewhere. In other embodiments of the present invention indices may be used to represent the actions in the data structure.

[0053] Fig. 5 shows a flow chart of a method for generating a scanner according to an embodiment of the invention described in the application of applicant entitled "Dynamic Scanner," referenced above, using start states associated with patterns to be scanned for. At least one respective start state of a plurality of start states is associated with each of the plurality of patterns (Step 502), the plurality of patterns being defined in Step 402 of the method depicted in the flowchart of Fig. 4 and discussed above. The plurality of start states may form part of the scanner definition data structure which includes the patterns and associated executable actions, as described above. A subset of the set of start states may be associated with each pattern in the scanner definition data structure.

[0054] At least one respective start state is processed along with each associated pattern so as to form a part of the scanner data structure (Step 504). The processing of Step 504 is performed as part of the processing Step 406 of the method depicted in the flowchart of Fig. 4 and discussed above. A current start state is set, the current start

state being one of the plurality of start states (Step 506). The comparing portion of Step 406 is performed so as to compare only the patterns associated with a respective start state equal to the current start state. The start states may thus be used as a way to control when particular patterns are "active," i.e., a being scanned, or searched, for in the scanning process.

[0055] According to an embodiment of the present invention, the current start state may be set using a stack of the start states, the current start state being defined as the start state at the top of the stack of start states. The stack of start states supports the usual operations of a stack (push, pop, get top, set top, is empty). In addition, the complete stack may be copied to or from another variable. For example, the stack could be modified as a part of an execution of one of the executable actions. An initial value for the current start state may be set at the start of a scanning operation. By appropriate control of the stack, the current start state may be changed as the scanning process progresses.

[0056] In some embodiments of the present invention, instead of "pure" start states, contexts may be used to control when a given pattern is active for searching by the scanner. Contexts are described in the application of applicant entitled "Dynamic Scanner," referenced above. A context includes rules defining where the context starts and ends. Starting a context means pushing the corresponding start state onto a start state stack. Ending the context means popping the start state from the stack. Among other advantages, using contexts prevents a user trying to pop a start state from an empty stack.

[0057] The text stream filter according to the present invention may find application in any suitable application in which a text stream filter may be employed, including in a parser filter, a parser, and a compiler, for example.

[0058] The present invention has been described herein with reference to specific

exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the claims that follow. The specification and drawings are accordingly to be regarded in an illustrative manner rather than a restrictive sense.